NPS55-78-030

NAVAL POSTGRADUATE SCHOOL

Monterey, California

A TARGET SELECTION MODEL

by

Gilbert T. Howard

October 1978

Approved for public release; distribution unlimited.

NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA

Rear Admiral T. F. Dedman
Superintendent

Jack R. Borsting
Provost

Reproduction of all or part of this report is authorized.

This report was prepared by:

ssc

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NPS55-78-030	2. GOVT ACCESSION NO. AD-A070083	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A Target Selection Model		5. TYPE OF REPORT & PERIOD COVERED Technical
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Gilbert T. Howard		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93940		12. REPORT DATE October 1978
		13. NUMBER OF PAGES 25
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Target Selection
Dynamics Programming
Job Shop Scheduling
Investigation Theory

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A target selection model is discussed in which the decision maker must select from a set of targets the subset having maximum total value. The targets are presented in a fixed order known in advance to the decision maker, but the choices are related so that not all selections are possible. Optimal decision of a target set among several decision makers is considered and formulated as a dynamic programming problem. The problem with two decision makers is used to illustrate a formulation which leads to a more efficient computational scheme. An example is included.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

1. INTRODUCTION

This report considers a problem of selecting from a set of objects a subset of maximum value where there are sequence dependent constraints which restrict the choices available. There is a natural ordering among the objects which dictates the order in which they must be considered. It is assumed that the decision maker has complete knowledge of the opportunities which will arise so that the problem is one of determining which opportunities to select and which to forego in preparation for later gain. The problem in which the opportunities have some probability of vanishing without notice is considered briefly, but the solution method is the same as long as these probabilities are known.

The problem can be cast in several forms. Reference 1 describes the problem in the context of an "investigator" passing through a region in which there is a number of points to be examined. It can also be viewed as a "delivery" problem in which a delivery truck has a list of locations and delivery times. If a delivery is made, it must be done at the given time and the specified location. Knowing the transit times from point to point and the value of each customer, the problem is to select the most valuable subset of customers to serve.

The problem can be viewed as a job shop scheduling problem in which the production manager has a set of possible jobs to perform. Each job has a required starting time and known set-up and processing time. The problem is to select the most valuable subset for processing. Reference 2 deals with the application of dynamic programming to a similar problem in which each job has an availability interval within which it must be processed if selected.

The problem is viewed here as one in which a defender is guarding some region against attack from a set of attackers whose times and points of arrival at the region are known. The defender is constrained to remain on the boundary of the region and must move to the point of intrusion at the time the attacker arrives in order to destroy the attacker. This is equivalent to saying that the defender can not engage the target until it crosses some threshold but it must be engaged immediately thereafter. The selection problem arises because the defender is limited in the rate at which he can move along the boundary.

For convenience the boundary is assumed to be a straight line segment along which the defender moves. Other configuration are possible. For example, some problems in point defense can be formulated in the same way. In that case the boundary can be thought of as a circle (of zero radius) and the position of the defensive system is represented by its angular rotation or orientation. Figure 1 illustrates a problem of this type when the region defined is the portion of the x-axis from 0 to B.

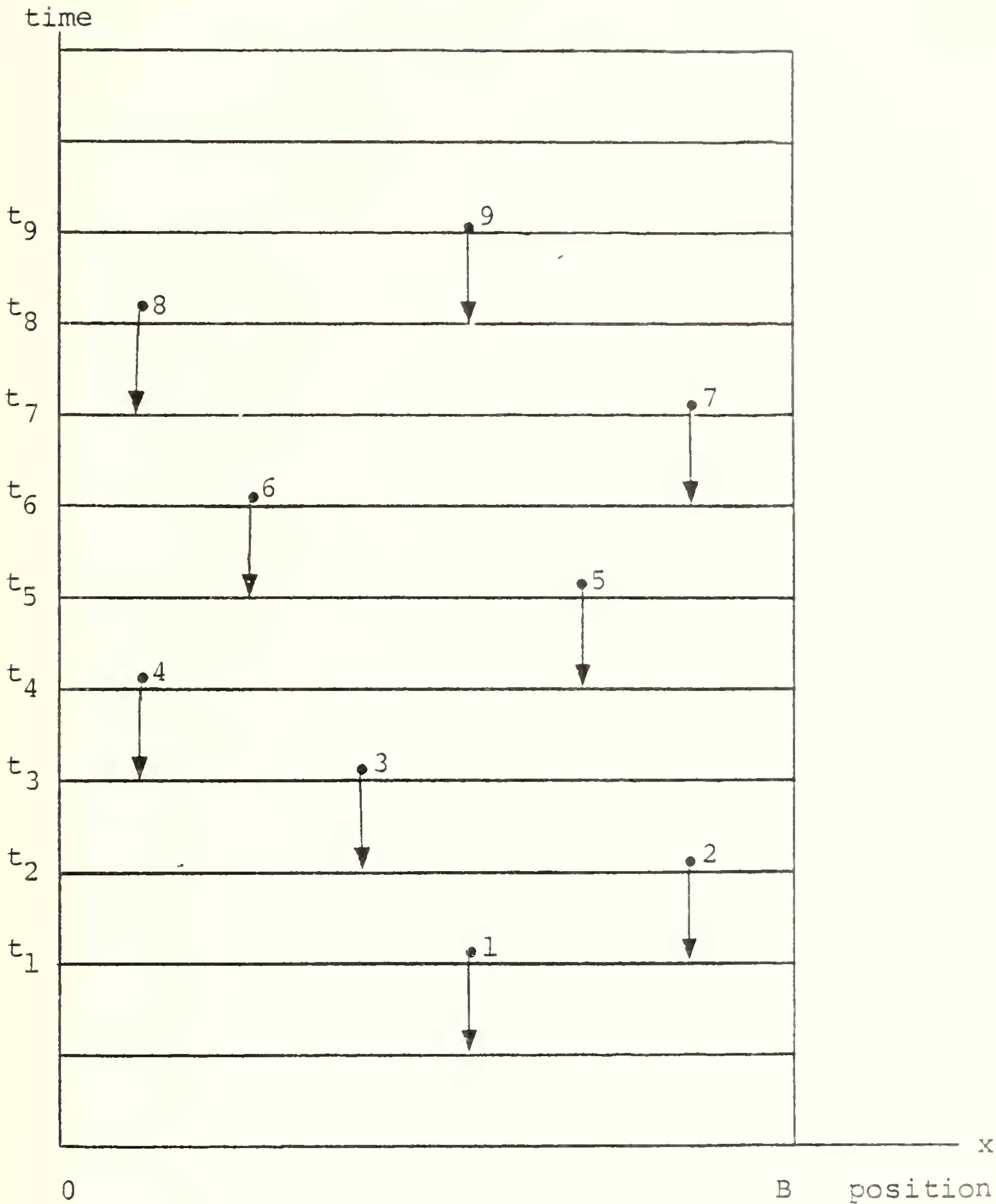


FIGURE 1. Illustration of Basic Problem

It is convenient to think of all the attackers as having the same speed and travelling on parallel courses directly toward the boundary, but it is not necessary to do so. In fact, all that is required is the time and point of arrival on the boundary for each attacker. It is assumed hereafter that the attackers are numbered in order of arrival at the boundary and that the times and points of arrival are known. Let t_j be the time of arrival of attacker j and p_j be the point of arrival where $0 \leq p_j \leq B$. The value of attacker j is represented by v_j. It is also convenient to define a dummy point p_0, t_0 with value $v_0 = 0$ representing the initial location of the defender.

For visual presentation of the problem a slight transformation is convenient. This problem is equivalent to one in which the attackers are stationary and the boundary moves in time through the attackers. As an example, the defender's position in the two-dimensional space (x,t) is shown in Figure 2.

The problem remains one of determining the optimal set of targets for the defenders to engage subject to limitations on his ability to reposition his defensive system.

This report deals primarily with the problem in which there are two identical defenders although the ideas are applicable for the M-defender problem. A straightforward dynamic programming approach for the M-defender problem is discussed first. See also Reference 1. Next an efficient

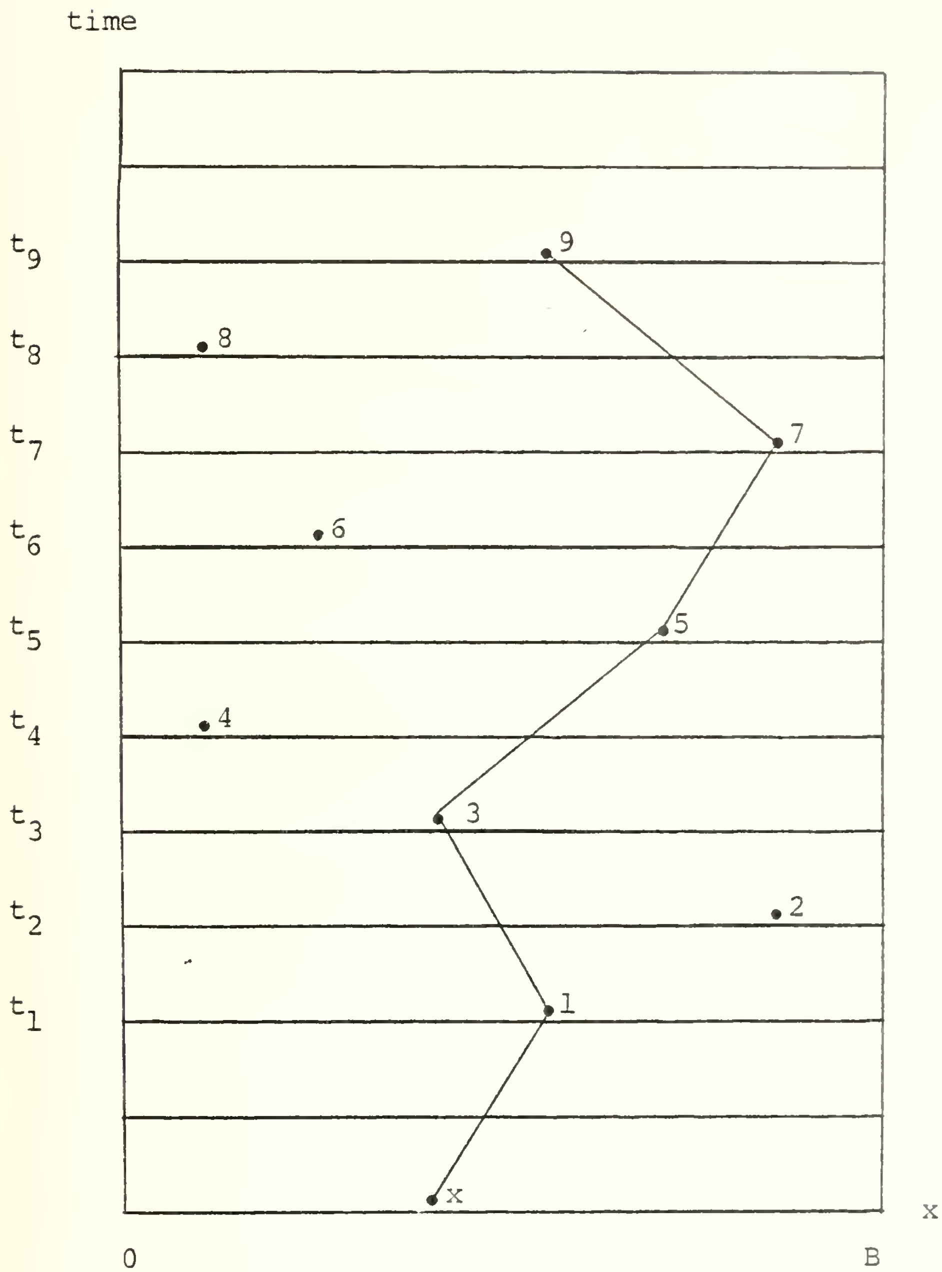


FIGURE 2. Space-Time Representation of Defender's Position

solution method for the problem with a single defender is considered. Then an improved formulation for the two-defender problem is presented and it is shown how that problem can be solved with a greatly reduced computational burden. A computer problem was written to test the method and the solution to a sample problem with $N = 50$ attackers is included.

Several generalizations are discussed in Section 5.

2. DYNAMIC PROGRAMMING SOLUTION FOR M-DEFENDERS

The M investigator problem can be solved with a standard dynamic programming approach using M state variables. Define the stages such that stage n corresponds to the time t_n, and let $\bar{X}_n = (x_{1n},\ldots, x_{Mn})$ be the state variable describing the location along the boundary of each investigator at stage n. The return function for stage n is

$$r_n(\bar{X}_n) = \begin{cases} v_n & x_{in} = p_n \text{ , any } i \\ 0 & \text{otherwise.} \end{cases}$$

The stage transformation functions can be written as

$$\bar{x}_{n+1} = \bar{t}_n(\bar{x}_n, \bar{D}_n)$$

where $\bar{D}_n$ is the vector of decisions prescribing the heading of all investigators as they move from stage n to $n+1$ and t_n is the function which describes the resulting locations at stage $n+1$. The decisions $\bar{D}_n$ are restricted to lie in some set K_n. An alternative formulation simply defines $\bar{D}_n$ as the set of points to which the investigators move so that $\bar{x}_{n+1} = t_n(\bar{x}_n, \bar{D}_n) = \bar{D}_n$, but still $\bar{D}_n$ is constrained to lie in some set of feasible solutions, say K_n.

The complete problem can be written as

$$\text{maximize} \quad \sum_{n=0}^{N} r_n(\bar{x}_n, \bar{D}_n)$$

$$\text{subject to} \quad \bar{x}_{n+1} = \bar{t}_n(\bar{x}_n, \bar{D}_n), \qquad n = 0, \ldots, N$$

$$\bar{D}_n \subset K_n, \qquad n = 0, \ldots, N$$

and can be solved using a standard dynamic programming approach with tabular computations once a suitable grid has been established for the state and decision variables. The solution requires the evaluation of the following recursive equations beginning with stage N

$$f_{N+1}(\bar{x}_{N+1}) = 0$$

$$f_n(\bar{x}_n) = \max_{D_n \subset K_n} \{r_n(\bar{x}_n, \bar{D}_n) + f_{n+1}(x_{n+1})\}, \quad n = 0, \ldots, N,$$

where

$$\bar{x}_{n+1} = t_n(\bar{x}_n, \bar{D}_n).$$

This method is very general and can be used to solve several variations of the basic problem, except that it becomes computationally unwieldy as M increases. The amount of computation required rises exponentially with M but linearly in N.

Two generalizations that can be solved using a similar formulation are the problems in which there is a restriction for each defender on the total number of attackers it can engage or a restriction on the total lateral movement of the defender. In these problems the vector of state variables must be enlarged to include a component for the number of engagements remaining and a component for the amount of lateral movement remaining.

The formulation above is not practical for large values of M nor for problems with constraints which lead to additional state variables, and the purpose of this report is to present a method which is more efficient than that just given, but first an efficient method for the basic one-state-variable problem is presented.

3. EFFICIENT SOLUTION FOR ONE DEFENDER

For the problem with a single defender, as illustrated in Figure 1, the algorithm shown in Figure 3 provides an efficient solution procedure. That algorithm assumes a dummy point at p_0 having value $v_0 = 0$ representing the starting location of the defender. The computation produces in reverse order the sequence of optimal return functions f_n, $n = 0,\ldots,N$ where the quantity f_n represents the optimal total return that can be obtained from the remaining stages, not including n, given that the position of the defender is p_n at time t_n. The sequence of optimal decisions is also produced and can be used beginning at d_0 to trace the optimal policy. The algorithm relies on the fact that the only relevant value of the state variable at any stage is the value corresponding to the location of the attacker and that the return functions need not be computed for any other values. For that reason the state variable is suppressed in f_n.

This algorithm can easily be implemented in FORTRAN as two nested DO loops.

4. IMPROVED FORMULATION FOR THE TWO-DEFENDER PROBLEM

This section shows how an improved formulation for the problem with two defenders can be obtained. In this section the N-attacker problem is still viewed as an N stage dynamic programming problem where stage n corresponds to the time t_n at which the defender departs from the target n. In this

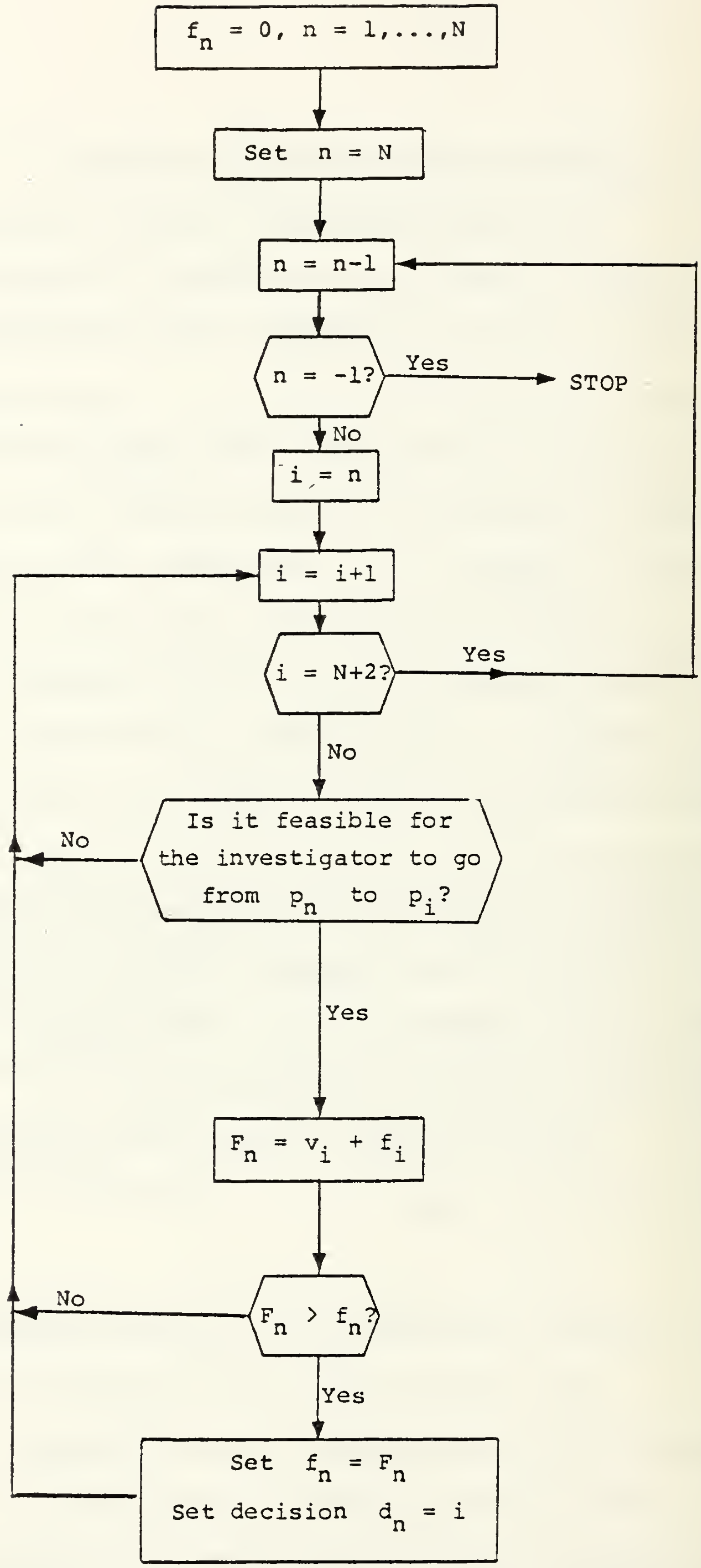


FIGURE 3. Algorithm for the problem with a single defender

formulation, as in the formulation of Section 3, passing through stage n corresponds to engaging target n; and if target n is not engaged, then stage n is bypassed.

At stage n the state variables x_{n1}, x_{n2} represent the positions of defenders 1 and 2 along the boundary. For computations it is assumed that a grid is established for the state variables. For convenience this grid is assumed to be the same at each stage, consisting of $S+1$ points equally spaced a distance Δ apart. It is also convenient to assume that the targets occur at the grid points, although this is not essential. If they were located between grid points it could be assumed that a target is engaged if the defender passes within a distance r of the target, or the grid points could be redefined at each stage to include the target location.

One observation which considerably reduces the amount of computation is that if the defenders pass through stage n, then one of the defenders must be located at p_n, the location of the n^{th} attacker. Thus instead of $(S+1)^2$ possible values of the state variables x_{n1} and x_{n2} we need to consider only $(S+1) + (S+1) - 1 = 2S + 1$ values. The computational reduction from this observation is roughly equivalent to reducing the number of state variables by one.

A second observation can also be made. This is that there is an optimal solution in which the two defenders do not cross. This is similar to the observation in the travelling salesman problem that for problems with Euclidean distance the optimal tour need not cross itself. The validity of this observation can be established by noting that if a point of crossing occurs, the defenders could simply be renumbered so that number 1 remains on the left and number 2 on the right. This follows from the fact that the defenders are identical, and it would not hold if they were able to move at different rates along the boundary. It would also be invalid if a penalty were assessed for changing the direction of motion from left to right or vice-versa or if they had constraints on the number of attackers handled.

With this second observation the number of possible values of the state variables is reduced even further. The possible values of the state variables at stage n, x_{n1} and x_{n2}, are given in Table 1 where p_n is the x-coordinate of the n^{th} target. Since the targets were initially assumed to lie on the grid points $p_n = K\Delta$ for some K. The state variable combinations listed in Table 1 have been numbered for reference, but notice that knowing the state number is equivalent to having the value of both state variables. For example, state number K' implies that

$$x_{n1} = \begin{cases} K'\Delta \ , & \text{if} \quad K'\Delta < p_i \\ p_i \ , & \quad\quad K'\Delta \geq p_i \end{cases}$$

and

$$x_{n2} = \begin{cases} p_i\Delta \ , & \text{if} \quad K'\Delta < p_i \\ K'\Delta \ , & \quad\quad K'\Delta > p_i \ . \end{cases}$$

State Number		x_{n1}	x_{n2}
0		0	p_n
1		Δ	p_n
$\vdots$			
K-1		$(K-1)\Delta$	p_n
K		p_n	$(K+1)\Delta$
			$\vdots$
S		p_n	$(S+1)\Delta$

TABLE 1

Possible values of the state variable in the two defender problem.

Thus all the state variable information is conveyed by the state numbers and it is unnecessary to write both state variables. For this reason it is adequate to let the single state variable at stage n be the state number S_n so that $S_n = 0,1,2,\ldots, S$. The result of this second observation is to reduce the number of state variable combinations at each stage from 2S+1 to S+1.

In the dynamic programming formulation $f_n(S_n)$ denotes the maximum total return from the remaining stages $n+1,\ldots,N$ given that the defenders are at stage n in state S_n. The general recursion can be written as

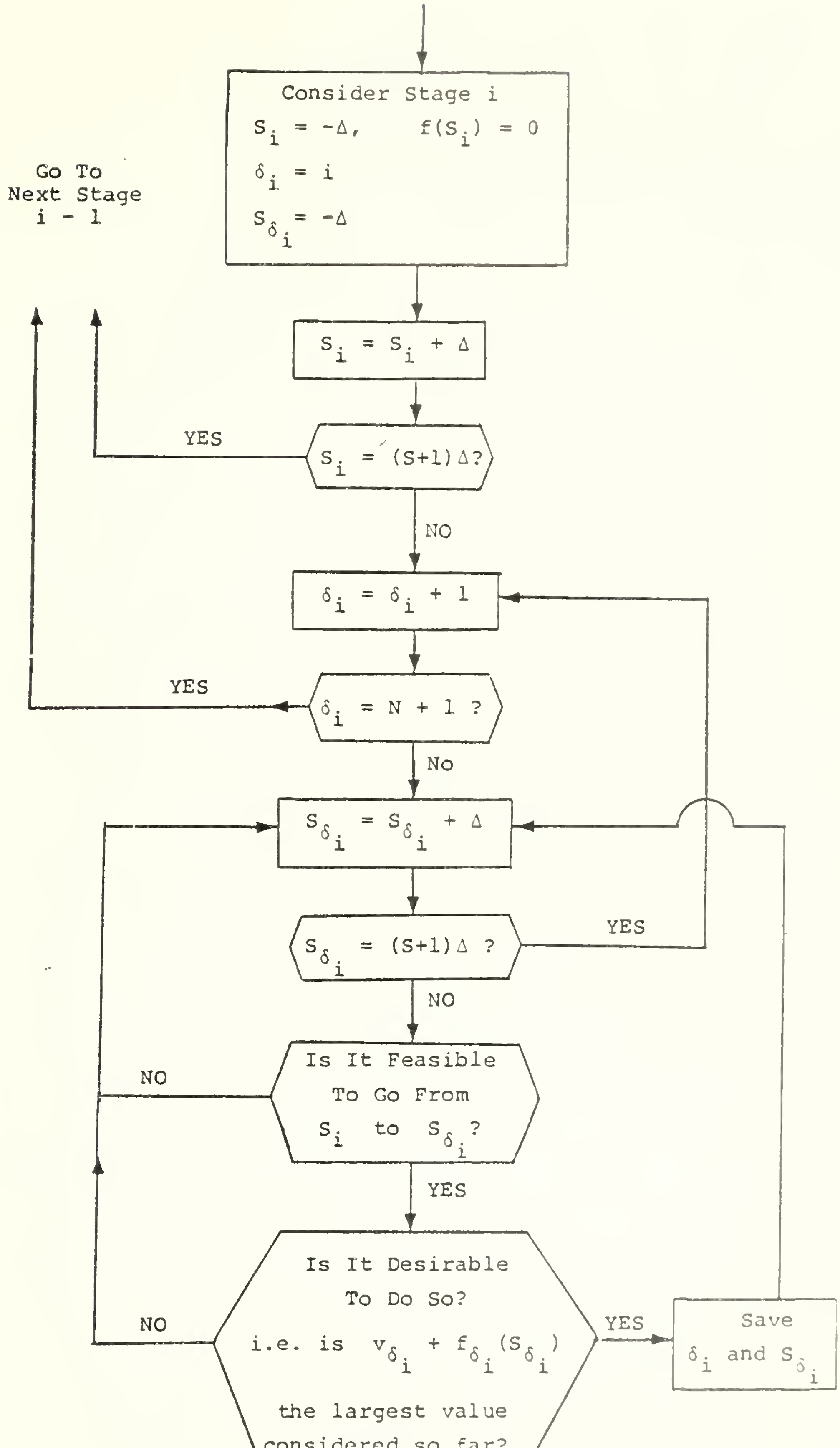
$$f_N(S_N) = 0$$

$$f_n(S_n) = \max_{\delta_n, D_n} \{v_{\delta_n} + f_{\delta_n}(D_n)\} ,$$

$$\delta_n > n,\ D_n \in K_{\delta_n}(S_n),\ n = 1,\ldots,N-1$$

where δ_n is a decision variable denoting the stage to which the defender moves next and D_n is the decision variable which determines the state number to which each defender moves at stage δ_n. The set of feasible states at stage δ_n given state S_n at stage n is denoted by $K_{\delta_n}(S_n)$.

From state S_n it may not be possible to go to some stages and, among those that are possible, the question of which to enter must be resolved. The general computation proceeds as shown in Figure 4 beginning at stage N-1. Figure 4 assumes completion of the computation of $f_n(S_n)$ for $n = N-1, N-2, \ldots, i+1$ so that the next step is to evaluate $f_i(\cdot)$.



Consider Stage i
$S_i = -\Delta$, $f(S_i) = 0$
$\delta_i = i$
$S_{\delta_i} = -\Delta$
Go To Next Stage i - 1
$S_i = S_i + \Delta$
YES
$S_i = (S+1)\Delta$?
NO
$\delta_i = \delta_i + 1$
YES
$\delta_i = N + 1$?
No
$S_{\delta_i} = S_{\delta_i} + \Delta$
YES
$S_{\delta_i} = (S+1)\Delta$?
NO
NO
Is It Feasible To Go From S_i to S_{δ_i}?
YES
Is It Desirable To Do So?
i.e. is $v_{\delta_i} + f_{\delta_i}(S_{\delta_i})$
the largest value considered so far?
NO
YES
Save δ_i and S_{δ_i}

FIGURE 4

5. MODIFICATIONS TO THE BASIC PROBLEM

A similar problem in which each target j has some probability $1 - p_j$ of vanishing before it is engaged can be solved by the same method. The objective becomes maximization of expected value and the computation is modified by diminishing the value of target j from v_j to $p_j v_j$. After this change, the remainder of the solution method is the same.

Several other modifications of the basic problem are of interest. These include the problem in which there is a limitation on the total motion of the defender as would be the case where fuel or some other resource is consumed. Another interesting problem arises when there is a constraint on the total number of attackers which can be engaged by each defender. Both of these modifications can be handled in the basic dynamic programming formulation by the addition of another state variable; but as mentioned previously large problems require that more efficient methods be found.

In the computer program which implements this solution method for the two-defender problem a further savings is incorporated. If the difference between δ_i and i is large, or at least if the time difference between the stages is large, it is almost certainly possible to engage some target between i and δ_i, say at stage j, and still move to any desired state at stage δ_i. In such a case it is unnecessary for the recursive procedure to examine stages $\delta_i, \delta_{i+1}, \ldots, N$ as possible choices of the next stage following i. This is because those choices are dominated by another choice, namely, moving to stage j from i. A test to this effect was incorporated with a considerable reduction in computer time. The usefulness of such a test depends on the particular rules of motion assumed for the defender and will be pursued here.

Reference 3 considers the problem from the point of view of two attackers moving through a region containing N targets. Each attacker must select a subset of the targets and no target can be selected by both attackers. The case in which the attackers enter the region from different directions is considered. That case is not covered here since the ordering of the stages differs for the two attackers.

6. COMPUTER PROGRAM

A program was written in FORTRAN IV to implement the procedure described in Section 4 for the two defender problem. This program was used to solve several problems in which the attacker's locations were generated randomly using a uniform distribution to determine the coordinates x and y, where $0 \leq x < 11$ and $0 \leq t \leq 100$. In order to make the x-coordinates fall on a conveniently spaced grid, the x-values generated were truncated to an integer before solving the problem.

Table 2 shows the x and t coordinates for a sample problem involving 50 attackers. Also shown are the randomly generated target values ranging from 1 to 5.

A graphical presentation of the sample problem is given in Figure 5. The numbers beside each point are the target values on the right and the point number (mod 10) on the left. The two lines show the sets of attackers engaged by the two defenders. The roles of motion used in this sample problem permitted the defenders to move left or right at the rate of one unit per unit time. The scale used in plotting Figure 5 is compressed in the t direction and it gives the appearance that the defenders can move more rapidly than one unit per unit time, but reference to the coordinates of each point will confirm that the rules of motion are not violated.

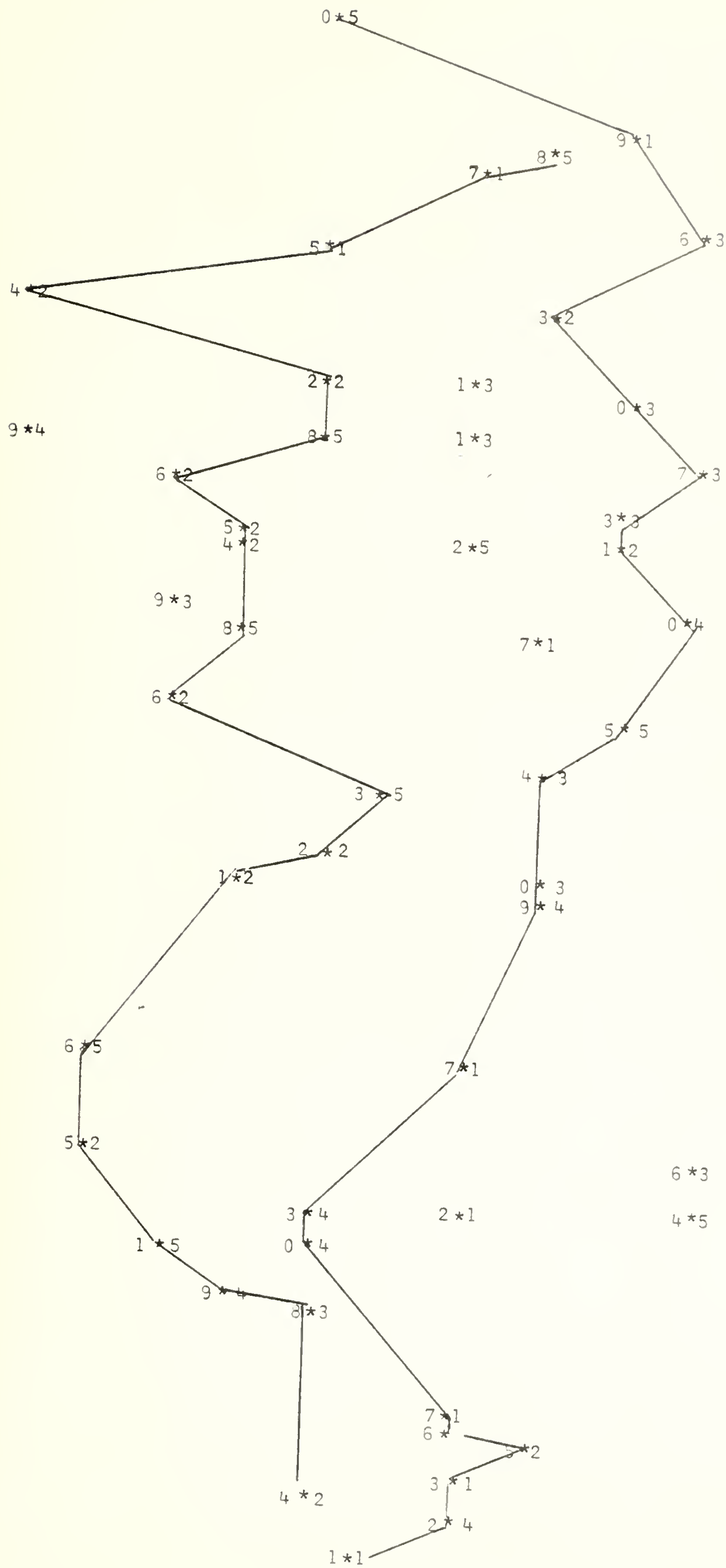

0*5
9*1
8*5
7*1
6*3
5*1
4*2
3*2
2*2
1*3
9*4
0*3
8*5
1*3
6*2
7*3
3*3
5*2
4*2
2*5
1*2
9*3
8*5
0*4
7*1
6*2
5*5
4*3
3*5
2*2
1*2
0*3
9*4
6*5
7*1
5*2
6*3
3*4
2*1
4*5
1*5
0*4
9*4
8*3
7*1
6*
*2
3*1
4*2
2*4
1*1

FIGURE 5

1	5.00	0.10	1
2	6.00	2.33	4
3	6.00	5.20	1
4	4.00	5.53	2
5	7.00	7.74	2
6	6.00	8.96	3
7	6.00	9.31	1
8	4.00	16.18	3
9	3.00	17.18	4
10	4.00	20.88	4
11	2.00	20.80	5
12	6.00	21.37	1
13	4.00	22.19	4
14	9.00	22.81	5
15	1.00	26.16	2
16	9.00	26.91	3
17	6.00	31.96	1
18	1.00	32.69	5
19	7.00	42.18	4
20	7.00	43.04	3
21	3.00	44.45	2
22	4.00	45.69	2
23	5.00	49.25	5
24	7.00	50.33	3
25	8.00	53.22	5
26	2.00	55.22	2
27	7.00	59.42	1
28	3.00	59.88	5
29	2.00	60.38	3
30	9.00	60.63	4
31	8.00	65.35	2
32	6.00	65.54	5
33	8.00	66.17	3
34	3.00	65.79	2
35	3.00	66.89	2
36	2.00	69.05	2
37	9.00	70.86	3
38	4.00	72.45	5
39	0.00	73.54	4
40	8.00	75.19	3
41	6.00	76.55	3
42	4.00	76.92	2
43	7.00	80.05	2
44	0.00	81.85	2
45	4.00	84.41	1
46	9.00	85.02	3
47	6.00	90.00	1
48	7.00	90.34	5
49	8.00	92.90	1
50	4.00	99.97	5

Table 2. Data for Sample Problem Involving 50 Attackers

REFERENCES

1. Gilbert T. Howard, Minimizing the Number of Penetrations in a Boundary Defense Problem, NPS55HK72041A, April 1972.

2. Charles F. Martin and Robert S. Poling, Fast Dynamic Programming Selection Algorithms for the Job Shop Problem with Availability Intervals, Paper presented at Western Section ORSA meeting, San Diego, California, Sept. 1977.

3. G. R. Hamelin, A Two-Directional Target Optimization Model Using the Branch and Bound Algorithm with Dynamic Programming, in preparation.

INITIAL DISTRIBUTION LIST

	COPIES
Defense Documentation Center Cameron Station Alexandria, VA 22314	2
Library, Code 0212 Naval Postgraduate School Monterey, CA 93940	2
Dean of Research Code 012 Naval Postgraduate School Monterey, CA 93940	1
Library, Code 55 Naval Postgraduate School Monterey, CA 93940	1
Center for Naval Analysis 1401 Wilson Boulevard Arlington, VA 22209	1
Naval Research Laboratory Washington, D.C. 20390	1
Naval Ocean Systems Center San Diego, CA 92132	1
Office of Naval Research Code ONR-230 800 North Quincy Street Arlington, VA 22217	1
Naval Surface Weapons Center Dahlgren, VA 22448	1
Naval Postgraduate School Monterey, CA 93940	
Attn: G. G. Brown, Code 55Bw	1
R. N. Forrest, Code 55Fo	1
J. K. Hartman, Code 55Hh	1
G. T. Howard, Code 55Hk	10
R. J. Stampfel, Code 55	1